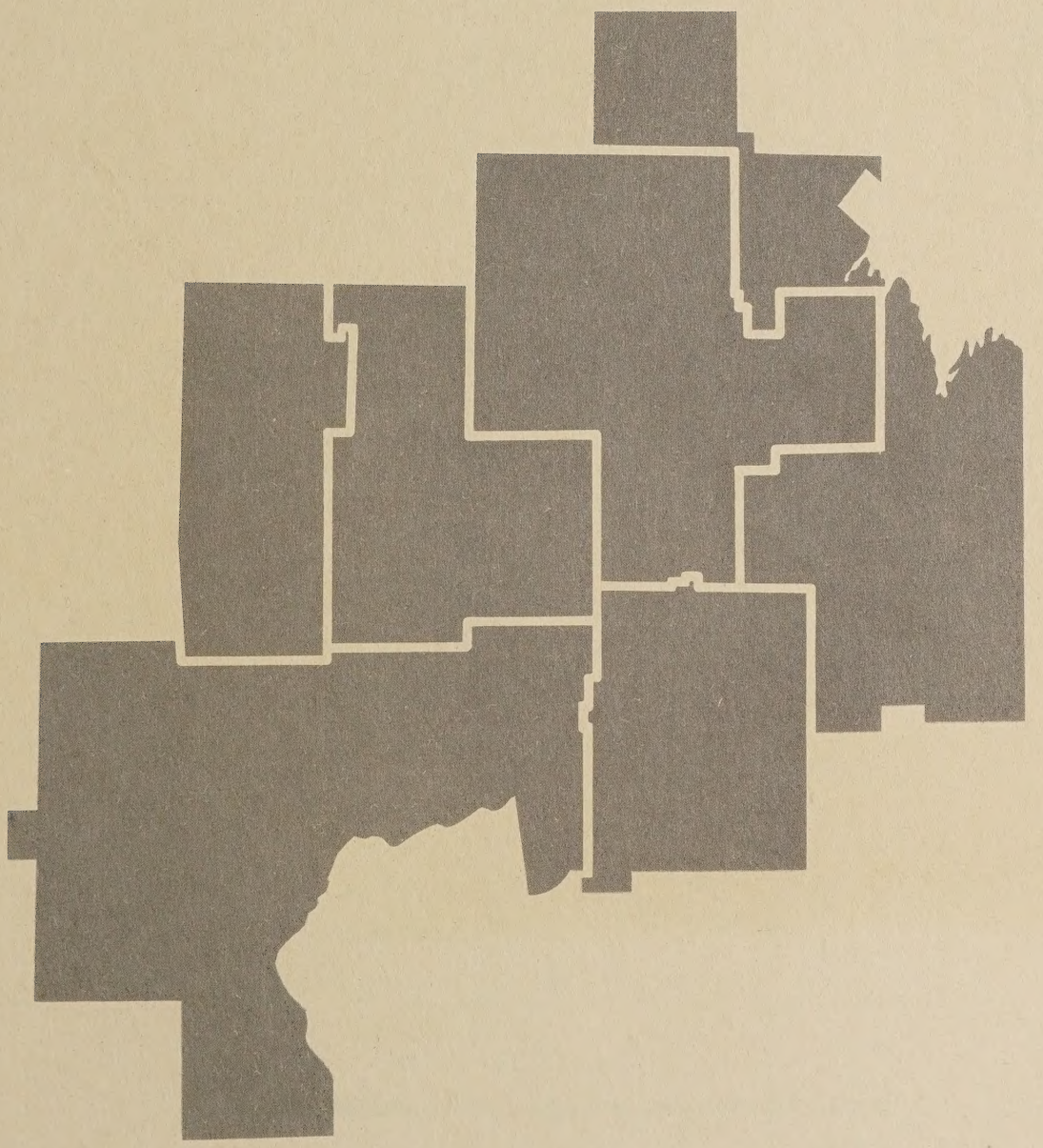
SCHEDULE 'D'

Computer Aspects



SUDBURY
AREA PLANNING STUDY

SUDBURY AREA
PLANNING STUDY

SCHEDULE 'D'

COMPUTER ASPECTS

JANUARY, 1974

THE STAFF

OF

THE SUDBURY AREA PLANNING STUDY

Mark Heitshu	Director
Pierre Beeckmans	Associate Director
Sylvia Cornell	Research assistant
Madelaine Vezina	Secretary
Rick Fujiwara	Graphics

GOVERNMENT CONSULTANTS

Michael Gain & Kevin Ryan	Computer Services Centre, Ontario Ministry of Treasury, Economics and Intergovernmental Affairs
Malcolm MacLoed & Frank Holt	Photogrammetry Office, Ontario Ministry of Transport and Communications

PRIVATE CONSULTANTS

M.M. Dillon Limited	Waterworks Sewerage Transportation Utilities Waste Disposal Flooding & Drainage
Gartner Lee Associates Limited	Terrain Evaluation Groundwater Septic Tank Suitability
Hough-Stansbury & Associates Limited	Wildlife Fisheries Vegetation Scenic Qualities Recreation Computer Program

TABLE OF CONTENTS

FIGURES

I

INTRODUCTION

OVERVIEW

The prime goal of the Sudbury Area Planning Study is to identify the areas in the study most suitable for further urbanization. All urbanization involves costs: either engineering monetary costs or environmental value costs. Each of these costs can be expressed as a constraint against development and the relative merits of different areas for urbanization can be described in terms of the degree of constraint existing in each area.

A methodology was devised to collect, organize, display and evaluate all the data that could be interpreted in terms of constraints. A summary of the computer-related activities is presented below:

(1) Preparation of inventory maps and constraint maps covering the study area.

(2) Transfer of data from the maps to a machine readable form.

(3) Transformation of the above data into a grid system.

(4) Rating and weighting of the constraint maps.

(5) Production and display of composite constraint maps.

ALLOCATION

Three separate agencies were involved in the computer aspects of the Study. Their responsibilities were allocated as follows:

(1) The Ministry of Transportation and Communications' Photogrammetry Section was responsible for digitizing the maps, that is, transforming the information on the maps into digital form and storing the digital information on magnetic tape for further processing.

(2) The Ministry of Government Services, Computer Services Centre was responsible for reading the digital information and transforming it into gridded information and storing it on magnetic tape for further processing.

(3) The consulting firm of Hough, Stansbury and Associates was responsible for reading the gridded information, forming composite maps and displaying them for visual evaluation.

The nature and extent of involvement of the three groups mentioned above is described in detail in the following pages.

II

DIGITIZING

1. GENERAL

A series of maps was produced by the Study staff and their consultants delineating the boundaries of all individual constraints against urbanization. Up to nine degrees or levels of constraints were shown on each map. Each map was submitted to the Photogrammetry Section of the Ministry of Transportation and Communications to be digitized.

Basically, digitizing consists of extracting information from the maps and reducing it to digital form and storing on a magnetic tape. In effect, the tape is a copy of the map and may, in fact, be used to reproduce the map.

2. TECHNIQUE

Inventory or constraint maps are laid out one at a time on a Gradicon digitizing table. A cursor with a set of cross hairs is electronically connected to a digitizing Unit with a magnetic tape drive. Digitizing consists of transforming the pictorial information from maps to machine readable magnetic tape. The digitizer may operate in either of two modes:

(1) point mode or (2) time mode.

In our study only time mode was used to contour the maps. The timing was set at 1 point every 0.7 seconds. As the digitizer operator moved the cursor over the areas which were delineated, the location of the cursor was recorded automatically in terms of X and Y every 0.7 seconds and the information was written directly out to magnetic tape.

Each map was digitized separately and in exactly the same format. Four geographical benchmarks in the study area were chosen as control points to locate, position and scale each map onto a common grid. The points in question were chosen because of their proximity to the boundaries of the study area. The points are described in the table below:

Description	Location
North	Benchmark nearest Capreol Lake
East	" " Appleby Corners
South	" " Ouellette
West	" " Worthington

The lower lefthand corner of each map was set to the relative reading X10000, Y10000. The four control points were then located individually by positioning a cursor with a set of cross-hairs immediately over the dot of the bench mark. The relative locations of the control points in terms of a Cartesian co-ordinate system were recorded separately on a magnetic tape by manually depressing a button each time.

Immediately following the four control points were the sets of vertices for each closed contour. (See Section III, Gridding, Fig. 5). Associated with each contour was a unique code to identify it and separate it from other contours. A tape mark was placed on the tape to indicate the end of a map. Several maps were placed on one tape. Data on the maps was recorded down to 0.001 inch. Any number of vertices were used to describe the shape of a contour as long as they were sufficiently close not to distort the shape.

The digitized information stored on magnetic tape was read by a mini computer where it was edited and corrected for the first time. The edited data was used (1) to drive an Automatic Drafting Machine and (2) to create an edited output tape. The Automatic Drafting Machine produced a copy of the contours which were delineated on the original maps. This provided a good check on the integrity of the digitized data. If any errors were detected, the output tape was edited a second time and the contours in question were redigitized and stored on a separate file to be merged with the original file by the Gridding Program. A block chart of this process is presented in the Appendix to Section III.

About 70 maps were prepared in total at three different scales.

- 50 maps at 1:250,000
- 2 maps at 1:100,000
- 18 maps at 1: 50,000

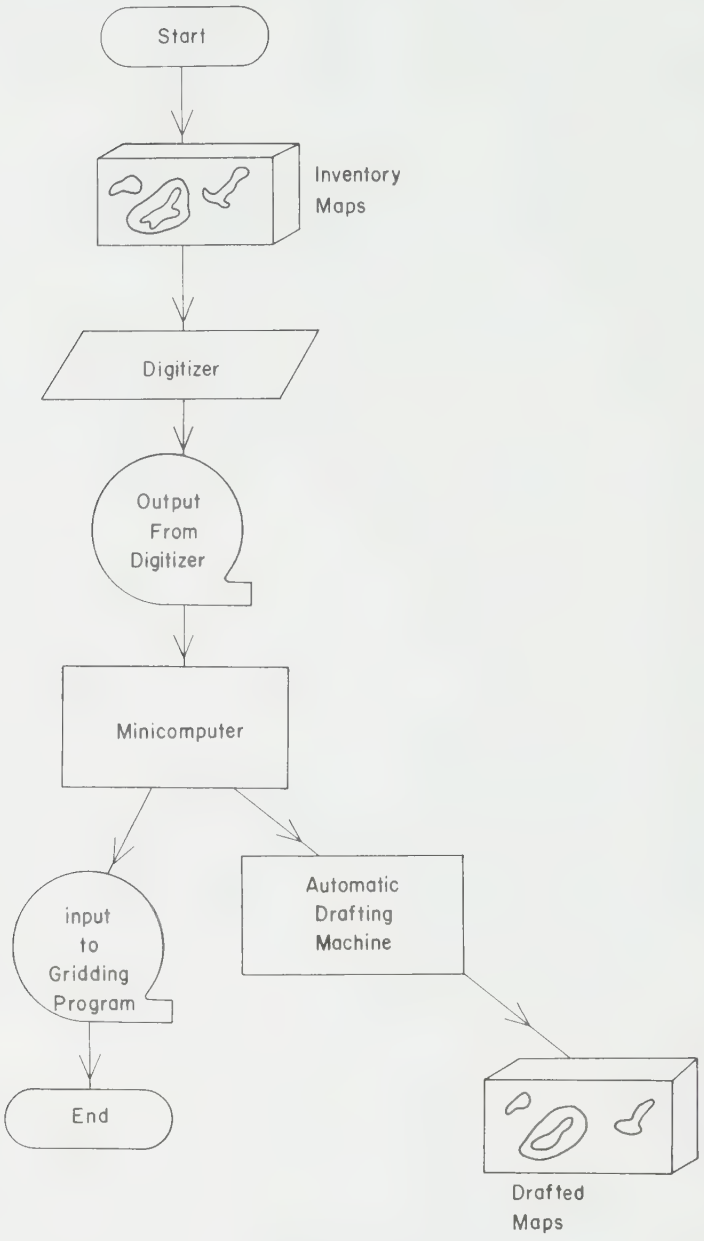
The 1: 50,000 maps were photo-reduced to about 1:100,000 to better fit on the digitizer table which has dimensions of about 4 feet x 3 feet.

3. SOFTWARE

Seven-track tapes produced by the Gradicon Digitizing unit must be converted to nine-track tapes to be compatible with the larger more powerful computer. The conversion from seven to nine-track tape is done on a Hewlett Packard 2116B computer using a program written by the Photogrammetry Office of the Ministry of Transportation and Communications. The program is loaded into the computer from a disk file via commands through a teletype terminal. Operator interaction through the teletype terminal is required to direct and control the operations performed by the conversion program. A flow chart of the system operation is presented in the Appendix to this Section.

FIGURE I

ABBREVIATED DIGITIZING PROCESS FLOW CHART



See also detailed flow chart in Appendix to Section II

III

GRIDDING

1. GENERAL

The magnetic tapes containing the digitized maps were passed on to the Computer Services Centre for further processing. To evaluate the overall effect of the factors at any point on the map, a grid was set up to cover the study area. It was determined that the final output from the computer would be a computer printout with dimensions 4' x 3'. This size was chosen as it corresponds to within a few inches of the dimensions for the maps of the Study Area drawn at 1:1000,000. On the final computer printout 1' equals 20 miles, and the area under each printed character represents about 22.2 acres. Since each printed character was 1/10th of an inch wide by 1/8th of an inch high, the number of characters that were printed across a page (columns) was 480 and the number of characters that were printed down a page (rows) was 288. Therefore, thinking of each print character as a cell, the entire Study Area was divided into 480 x 288 cells (i.e. 138,240 - 22.2 acre cells). (1)

The Computer Services Centre provided a gridding program to overlay each digitized map and to place in each 22 acre cell a code from 0 - 9. The code placed in each cell was determined from the data contained on the digitized tapes. Each map was processed individually. The control points were read in and transformed to fixed cells in the overall grid matrix. Using these cells as reference points along with the actual readings for the control points, vertical and horizontal scaling factors were determined for each map. A linear conformal transformation was performed on the vertices which described each contoured area to ensure that corresponding geographical areas from the various maps would all be placed in the same relative grid cells. The net result of the scaling and transformation was to put each geographical 22-acre cell from each map into the same relative location in the overall grid matrix each time.

The gridding program determined the code or value to be placed at the centre of each cell. The allowable codes range from 0-9. When all the cells were assigned their codes or values according to what was picked up from the digitized tape, a grid matrix or map matrix was the end result. The dimensions of each map matrix were 480 x 288 (columns x rows) or 4' x 3' corresponding to an area 80 miles x 60 miles. To ensure that the map was gridded correctly, the values or codes in each cell of the map matrix were printed out.

In addition each map matrix was stored on magnetic tape row by row. A number of map matrices were stored on a single magnetic tape. The end of each map was indicated by a tape mark.

(1) - Referred to as a 22 acre cell.

2. GRIDDING PROGRAMS

1. INTRODUCTION

The gridding package consisted of five programs and one utility. These programs were designed to run on the UNIVAC 1106 of the Computer Services Centre, Ministry of Government Services.

The first program reads a digitized data tape and produces a file containing the co-ordinates and other relevant information of both control points and points on contour boundaries within the map.

The second and third programs use this file to produce a grid matrix of the area, with the grid cells of the matrix containing values indicating whether or not they are inside a contour boundary. This grid matrix is transferred to an output file in unformatted format.

In the fourth program all the files are merged into one. This file is then used as input to the utility FSØRT which produces a sorted output file.

The final program blocks this data and converts it into a file which is IBM compatible. This file is for input to the weighting phase of the overall project.

2. DIGITIZED DATA PROGRAM

This program picks up information from the digitized data tape and creates a file for each map processed. Each record of the file consists of 27 characters and has the following format:

	Code	Sub Code	Accuracy Code		X-value		Y-value		Z-value
$				X		Y		Z	
(1)	(3)	(4)	(1)	(1)	(5)	(1)	(5)	(1)	(5)

Code - the codes for which this program creates records are '700' for control points and '202' for contour points, and '100' for drafting the subcode.

Sub-code - the first character of the sub-code indicates the inside level, the following three characters indicate the contour number.

Accuracy code - the accuracy code is a single digit not used in our processing.

X-value
Y-value - these are five-digit co-ordinates accurate to one-thousandth of an inch.

Z-value - the first three digits of this value are the map number, the last two are the outside level of the contour.

Input

The input file is on a 9-track tape in EBCDIC format. As well as control point information and contour point information, the file also contains data concerning the boundaries of the map, heading information and data records generated before each contour is started. This file is read through the EBCDIC hardware translator into the digitized data program.

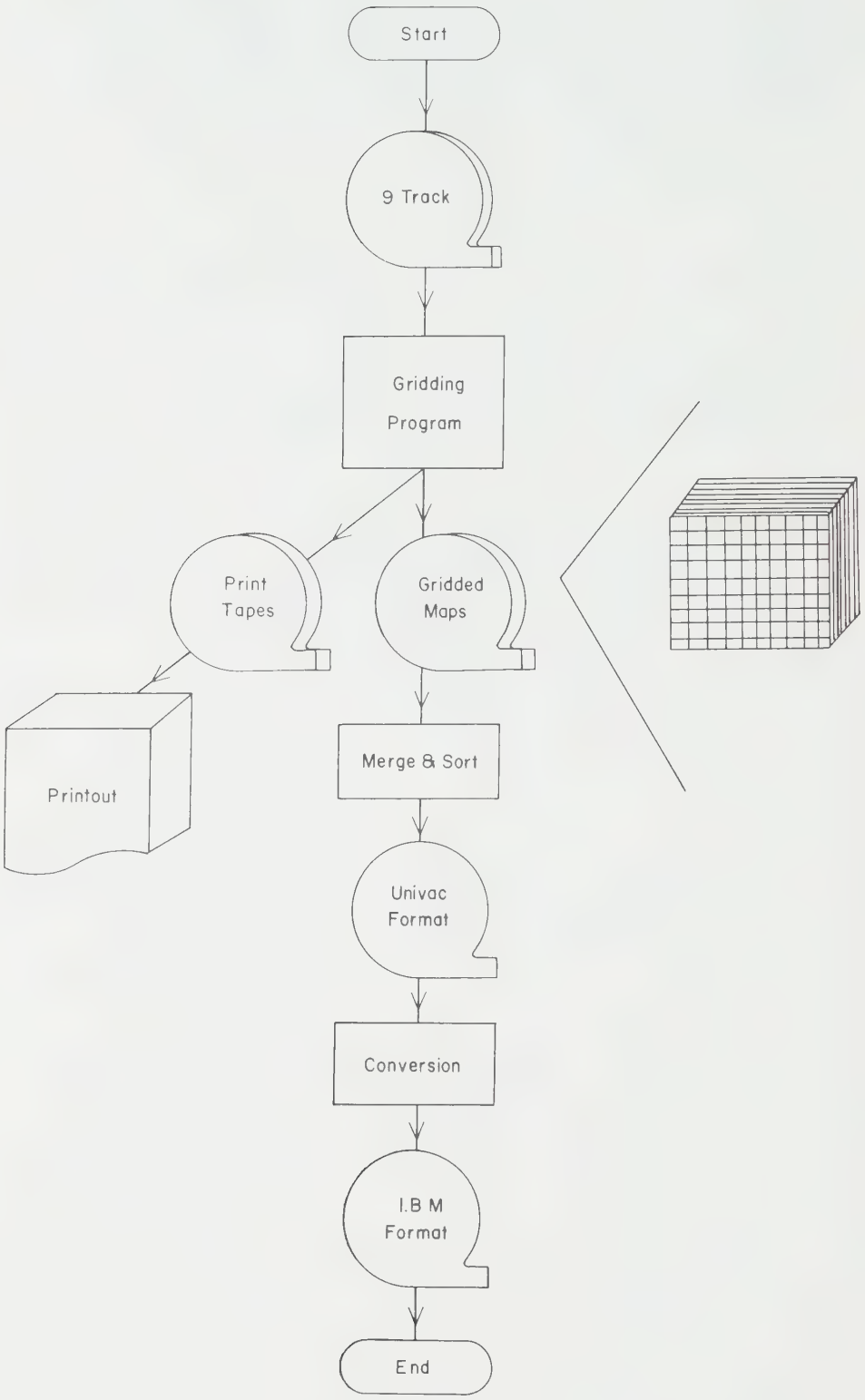
Method

This program is written in COBOL. The first four records written are in the control point records, they are the first four records in the input file. The program then searches through the input blocks writing only those records which have a code of '202' contour boundary points.

Output

The format of each record of the output file is as described above. The first four records have a code of '700' - control points, the remainder are contour boundary points and have a code '202'. The file is a RECORDING MODE 1 file so that it can be read by the FORTRAN coded grid programs.

FIGURE 2
ABBREVIATED GRIDDING PROCESS FLOW CHART



3. GRID PROGRAM 1

This program is used to create a grid (480 x 288) of cells. Each cell contains two values: values other than zero indicate that the cell lies within a contour boundary; the first value in the cell indicates the boundary number and the second value the level within the boundary. This particular program is used for the simple cases where there is only one level within any boundary. The matrix of cell values is maintained on direct access mass storage in this particular case, however, it could be maintained in core if sufficient core were available. The program can be broken into the following steps:

Step 1

Initialize all values of the grid matrix to zero.

Step 2

Read the input file. The input file is the output file of the DIGITIZED DATA PROGRAM and since it is a COBOL formatted file, it is read via the OGFIOC subroutine. The first four records are read to obtain the X and Y co-ordinates of the control points. From these control points, four constants are generated; these constants are used as operators on the co-ordinates of the boundary points to effect a conformal transformation so that all the maps are exactly overlayed. The constants are:

(1) XFAC - obtained by dividing the difference between X-east and X-west into 12578.

(2) XFAC1 - obtained by subtracting 16701 from the product of X-east and XFAC.

(3) YFAC - obtained by dividing the difference between Y-north and Y-south into 10351.

(4) YFAC1 - obtained by subtracting 22129 from the product of Y-north and YFAC.

(5) COSA - the cosine of the angle between the lines formed by joining X-west and X-east on the base map and on the particular map being analyzed.

(6) SIN A - the sine of the above angle.

Two other constants are also set at this stage of the program. P1 is put equal to 42.239; this allows the 480 columns of the matrix to represent 80 miles. P2 is put equal to 52.931; this allows the 288 rows of the matrix to represent exactly 60 miles. These constants are derived from the fact that the scale of the base map was 1:250,000 and that a computer print character is 1/10th of an inch high and 1/8th of an inch wide.

Step 3

Read the 'X' and 'Y' values of a single contour into an array, transforming the values as they are read. The maximum and minimum 'X' and 'Y' values are selected as the values are being read.

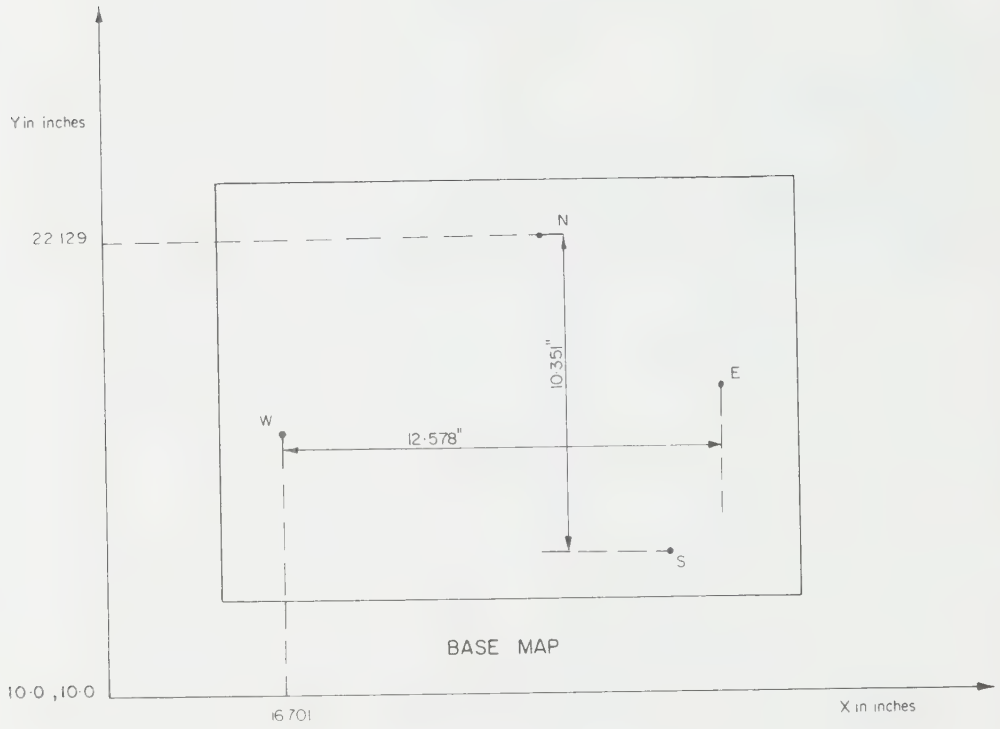


FIGURE 3

BASE MAP FOR SCALING
AND TRANSFORMATION

Step 4

Convert the maximum and minimum 'X' and 'Y' values to their appropriate column and row numbers. For example:

```
XCMAX = ((XMAX - 13000)/P1) + 1
YRMAX = ((YMAX - 10000)/P2) + 1
```

- the transformed origin is the point (13000,10000).

A limit check is also made at this stage to determine that the X-column values remain within the limits of 1 and 480, and that the Y-row values remain within 1 and 288.

Calculating these values allows us to examine a limited area within the total grid to determine whether points are inside or outside the contour boundaries. If, for example, we had the values:

```
XMIN  =  120
XMAX  =  245
YMIN  =   32
YMAX  =  163
```

We would apply the point-in=polygon technique only to those points within the inner boundary.

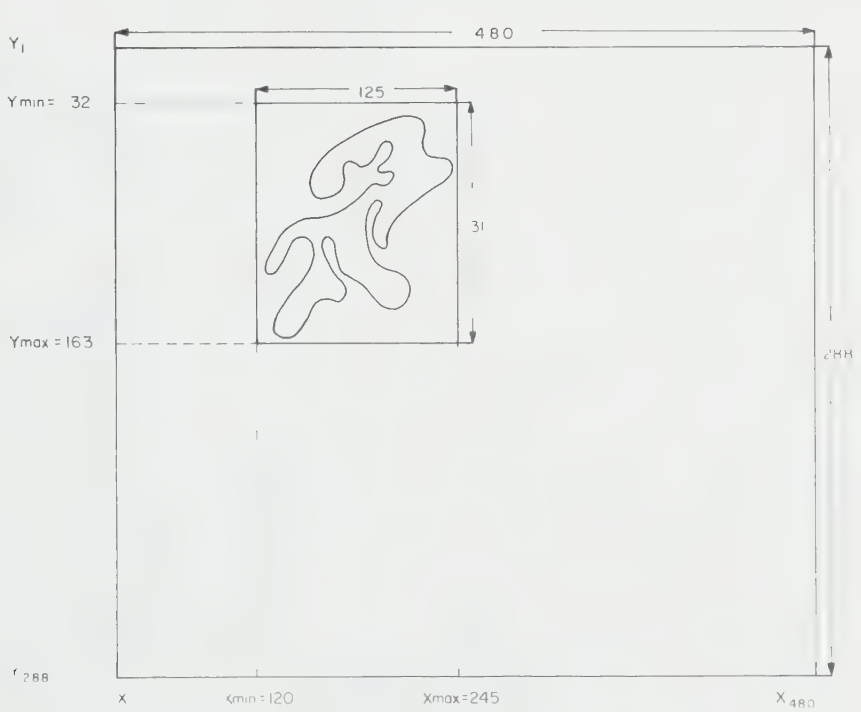


FIGURE 4

LIMIT SELECTION FOR
POINT IN POLYGON TECHNIQUE

The point-in-polygon technique is an algorithm for determining whether or not a point lies inside a polygon. The coordinates (X,Y) of the centre of each grid cell are known, as are the vertices (x_i, y_i, i = 1,N) of the polygon. Consider the diagram below:

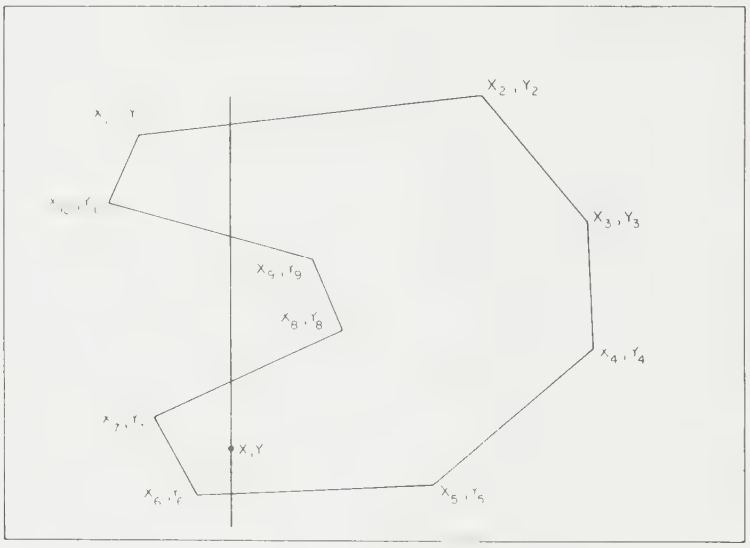


FIGURE 5

TYPICAL POLYGON

The algorithm has the following steps:

(a) Consider in turn each of the N lines bounding the polygon and determine if it crosses the X co-ordinate of the point.

(b) If it does cross the X co-ordinate of the point, calculate the Y value at the point of intersection of the polygon boundary line and the X co-ordinate line.

(c) If this Y value at the point of intersection is greater than the Y value of the point of observation, add 1 to a counter (CNT) which had been initially set to zero.

(d) When all the segments have been processed, if CNT is an odd number, then the point lies inside the contour boundary.

In the example above, the X co-ordinate intersects 4 lines of the boundary. However, in only three of these cases is the Y value at the point of intersection greater than the Y value of the point of observation and so CNT = 3 and the point lies inside the boundary - as verified by observation.

When a point is determined to lie within the boundary the value of the grid cell is set to the value of the inside boundary as read in with the records for that contour.

Step 5

When all the records in the file have been analyzed the output tape is created. The matrix at this stage is stored on a direct access file consisting of 288 records, each 480 words in length. Each record has two additional words prefixed before being written out to tape. The two words are the line number and the map number, this additional information is used for the sorting phase of this package.

Step 6

The final step is to produce a printout of the grid data. The printout is printed at 8 lines/inch, and hence the total final output is 4 feet by 3 feet - in exact proportion to the 80 miles x 60 miles being represented.

GRID PROGRAM 2

This program is a modification of the previous program and is used where we have a more complicated situation such as boundaries within boundaries with levels changing. The steps in the program are the same as those in the previous program except where we assign the level of the inside boundary to the grid cell found to be within the boundary.

The technique used here is to have a second matrix with exactly the same dimensions as the grid matrix. This second matrix is a switch matrix whose cells contain values which are used to determine the values of the grid matrix.

When a point is found to be within the boundary, the corresponding cell in the switch matrix has added to it (1 + inside level), and the lowest value (LFLIP) of all the cells changed is kept. When the rectangular area bounded by XMIN, XMAX and YMIN, YMAX has been completely analyzed, we reexamine each cell in this area of the switch matrix once again. Where we find that a cell in the switch matrix has a value equal to LFLIP, we set the corresponding cell in the grid matrix equal to the inside level of that boundary.

In the following examples we will assume that boundaries with the lowest inside level are processed first. This assumption is not necessary, it merely makes for uniformity with the two diverse examples.

5. MERGE AND SORT

The MERGE program reads all the individual files created by the GRID program and produces one output file. The program can also be used to add additional files to a previously created large file.

This merging of all the files into one is necessary for the sorting step which requires only one input file.

The sorting was done by means of a utility called FSØRT which will sort either a formatted or an unformatted file. In our case we had an unformatted file. The sort was performed on two keys: the major key being the record or line number and the minor key the map number. The output from this sort is a single file whose records have the same format as the input file.

6. CONVERSION

This program takes the output from the FSØRT routine, blocks it, and converts it to IBM - compatible EBCDIC format.

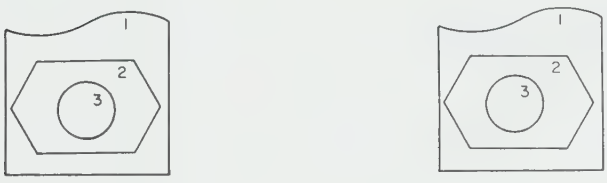
The records are read into a 17-record block, converted to character format from their initial unformatted state, and then passed through the EBCDIC hardware translator to produce the IBM-compatible file.

FIGURE 6

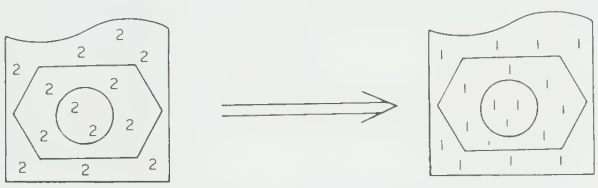
GRIDDING FROM OUTSIDE TO INSIDE

The levels are indicated in the first set of drawings only -

SWITCH GRID

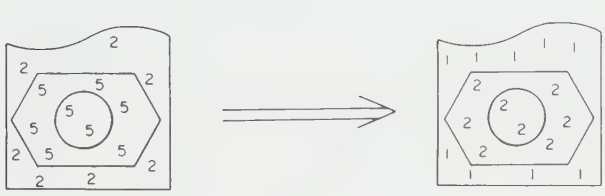


After analyzing inside level one boundaries -



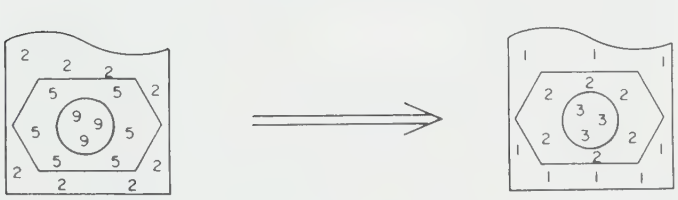
LFLIP = 2

After analyzing inside level two boundaries



LFLIP = 5

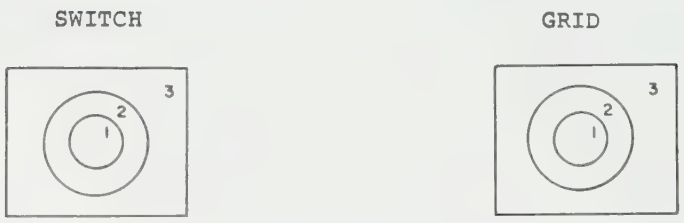
After analyzing inside level three boundaries -



LFLIP = 9

FIGURE 7

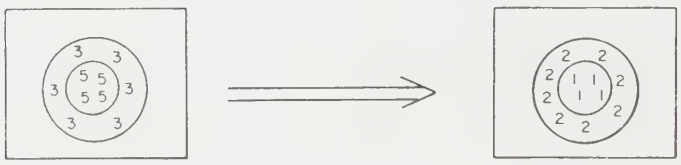
GRIDDING FROM INSIDE TO OUTSIDE



After analyzing inside level one boundaries -

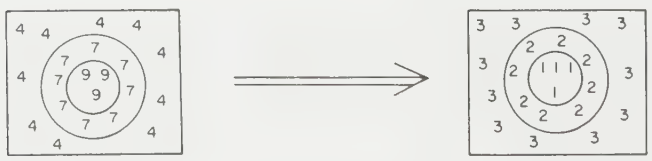
LFLIP = 2

After analyzing inside level two boundaries -



LFLIP = 3

After analyzing inside level three boundaries -



LFLIP = 4

This program could be used successfully to process boundaries with only one inside level. However, since the processing involved is greater, it is best to have a separate program for the simpler cases.

COMPUTATION AND DISPLAY

I. INTRODUCTION

The gridded representations of maps, having been merged and sorted, were made available by the Computer Services Centre to Hough, Stansbury and Associates Limited, as input to the manipulative package.

This package enables a user to rate numerically the different levels on any or all of the data maps and also to weight the whole map as a unit prior on adding up any specified maps to generate a composite. These rated and weighted maps can then be added in any combination to generate a composite score map. The program also keeps track of the highest and lowest occurring score-map values so as to enable ranking into levels further on. There is an automatic compilation of a table showing frequency of occurrence of levels in the composite. Other useful options included are:

(1) Title - the card following it is used for identifying the composites.

(2) *Windowing** - enables the user to restrict computation and display to a particular township, or alternatively to a rectangle with user specified co-ordinates.

(3) *Masking** - enables the user to specify logical operations of the type: If data value of a cell on Map #.....is Level..... and EQ/NE then the corresponding cell on the composites is replaced by the character '....'. For example, this would permit replacing all cells in the flood plain with a letter 'F'on any composite.

(4) Usage - could be one of 1. Display only
2. Datum to a further comparison or correlation (Display is implied)
3. Compare
4. Correlate
5. Compare and store as new datum
6. Correlate and store as new datum.

The full instruction-set of commands for the different programs is covered in the Appendix.

* Initial reference to a defined term is indicated in distinctive type to facilitate subsequent retrieval.

The manipulative package was developed with user convenience in mind. It was found that, for one complete run, a maximum of 1810 computer full words had to be specified every time. To minimize the clerical errors inherent in handling these 1810 full words (which will interchangeably be referred to as a *package* in this report), it was decided to develop a user-oriented strategy as follows:

(1) The user can copy any predefined package including the one that was saved just previous to the current package (called old) or the one which was loaded as a unique standard package. This brings the full package into core for further modification.

(2) The user can selectively make changes in this copied package to tailor it to his requirements.

(3) The instructions used in this program (to be referred to as *user instruction-preprocessor*) are based on user-oriented, meaningful, keywords, a list of which is appended to this report.

(4) The User-Instruction is immediately followed by error messages and diagnostics relating to this instruction, if any. This ties the Instruction-in-error to its corresponding diagnostics, thus making it easier to find and fix the error.

(5) A useful feature incorporated in this preprocessor is the command to save or not to save the package. This enables the user to assemble his instructions until they are error-free and to look at the interpretation of the instruction package before deciding to save it on the data set. It also helps in familiarizing the user with this program by means of comprehensive diagnostics.

After the saved package has been stored on the data set, the interpreted output assigns the package a unique identification number. The next task uses the command to execute a specified package number and the processing after this point is fully automatic and does not involve any further user intervention.

2. PROGRAM PACKAGE DESCRIPTION

There are four distinct functional requirements and tasks in the program package.

Task I To convert some of the map-matrices to conform to the requirements of succeeding programs (Clean-up and data aggregation). This mainly involves changing zero levels to non-zero levels, as the levels can then be used directly as a subscript variable. This is necessitated by the inability of Fortran to handle zero subscripts. When this feature is used, all levels in that particular map are incremented by one, so as to keep their relationships intact. This program also aggregates 51 original maps pertaining to environmental features into 5 category composites i.e. wildlife, fisheries, recreation, scenic, and vegetation. Task I is a programmer-oriented routine and is user-transparent. As it is a one-shot routine, it is run before the user is allowed access to the rest of the program package.

Task II The second functional requirement is to permit examination of the original data makeup. The program meeting this requirement prints the various data values for every cell within a specified window.

Task III The third task is the instruction-preprocessor with interleaved diagnostic generation and saving of error-free assembled packages and their interpretation. These saved packages are assigned unique identification numbers.

Task IV The fourth task relates to the activation of the actual run:

Task IV, Program 1 - the major computational program which generates the derived composites, does masking and replacement with specified characters, collects statistics on the basis of specified data originals and shows up changes in a series of generated composites on the basis of changes only or by correlation of specified levels. This program passes on the complete instruction-package and incidental new computed information for use by the last step in this task. A minor function of this program is to delete specified packages from the data set.

Task IV, Program 2 - is the IBM supplied sort/merge package.

Task IV, Program 3 - is the logical continuation of the first program in this task.

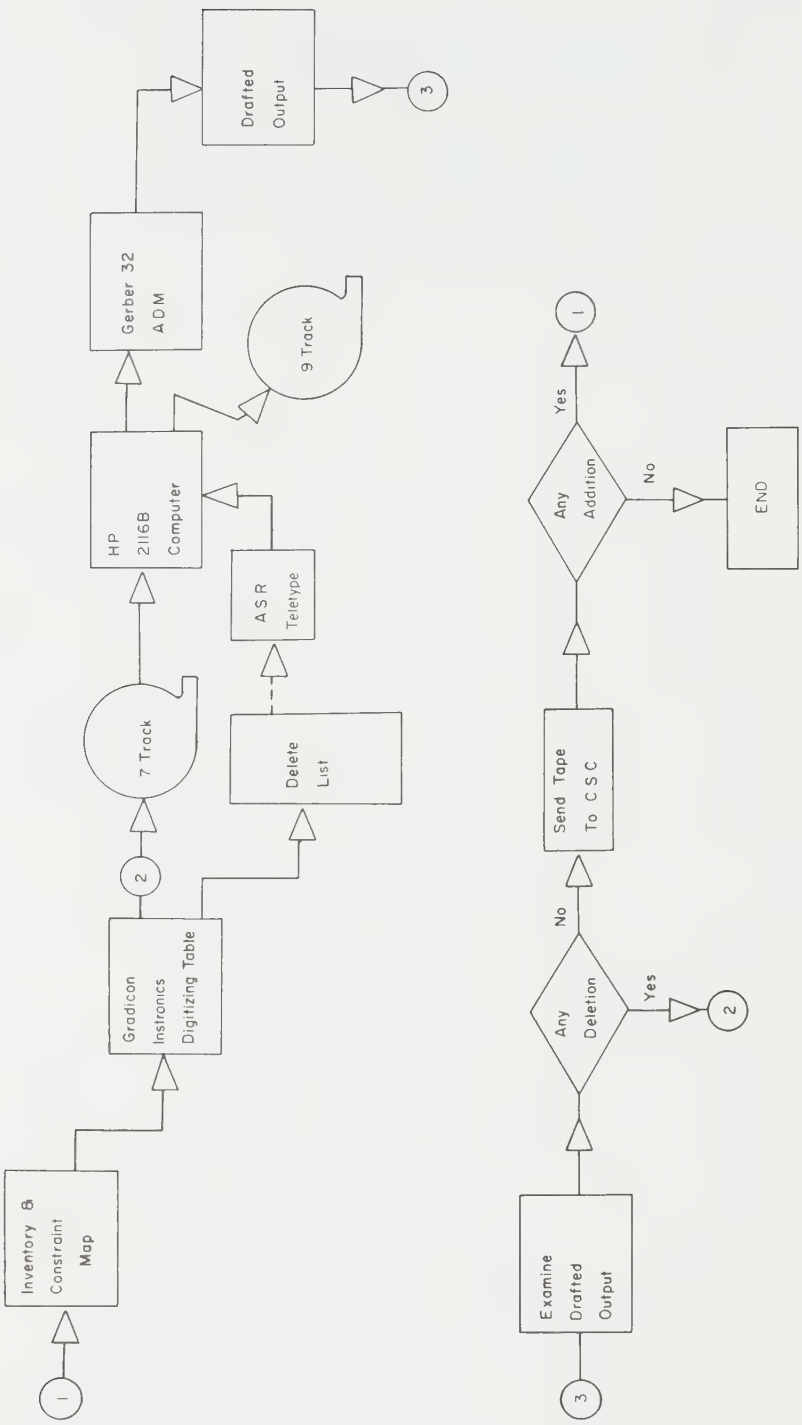
APPENDICES TO SECTION II

1. Digitizing Process Flow Chart

2. Instructions for 7 to 9 Track Conversion

3. Flowchart for 7 to 9 Track Conversion

1. DIGITIZING PROCESS FLOW CHART



2. INSTRUCTIONS FOR 7 TO 9-TRACK CONVERSION - EVEN RECORD/BLOCK

USE:

To convert 7-track (200, 556, 800 bpi) tapes on the M12 to 9-track (800 bpi) on the 7970.

Std	M12 Code	-	BCD
Std	7970 Code	-	EBCDIC
Std	M12 Buf Size	-	up to 1024 char
Std	7970 Buf Size	-	2000 char (2 char/word)
Std	File Search	-	one file/run

OPERATION:

Start 35000

- loads its own SIO base pg. link
- if a block (*) will overfill record, pads last of buffer with blanks
- pads last partial buffer with blanks
- if char count odd, pads second half of last word in buf with a blank
- HLT 64B at file mark; run with choice of any switch set gives 9 trk file mark, no switch run gives HLT 77B and no file mark i.e. next run merges another 7 trk tape

File Search 35217

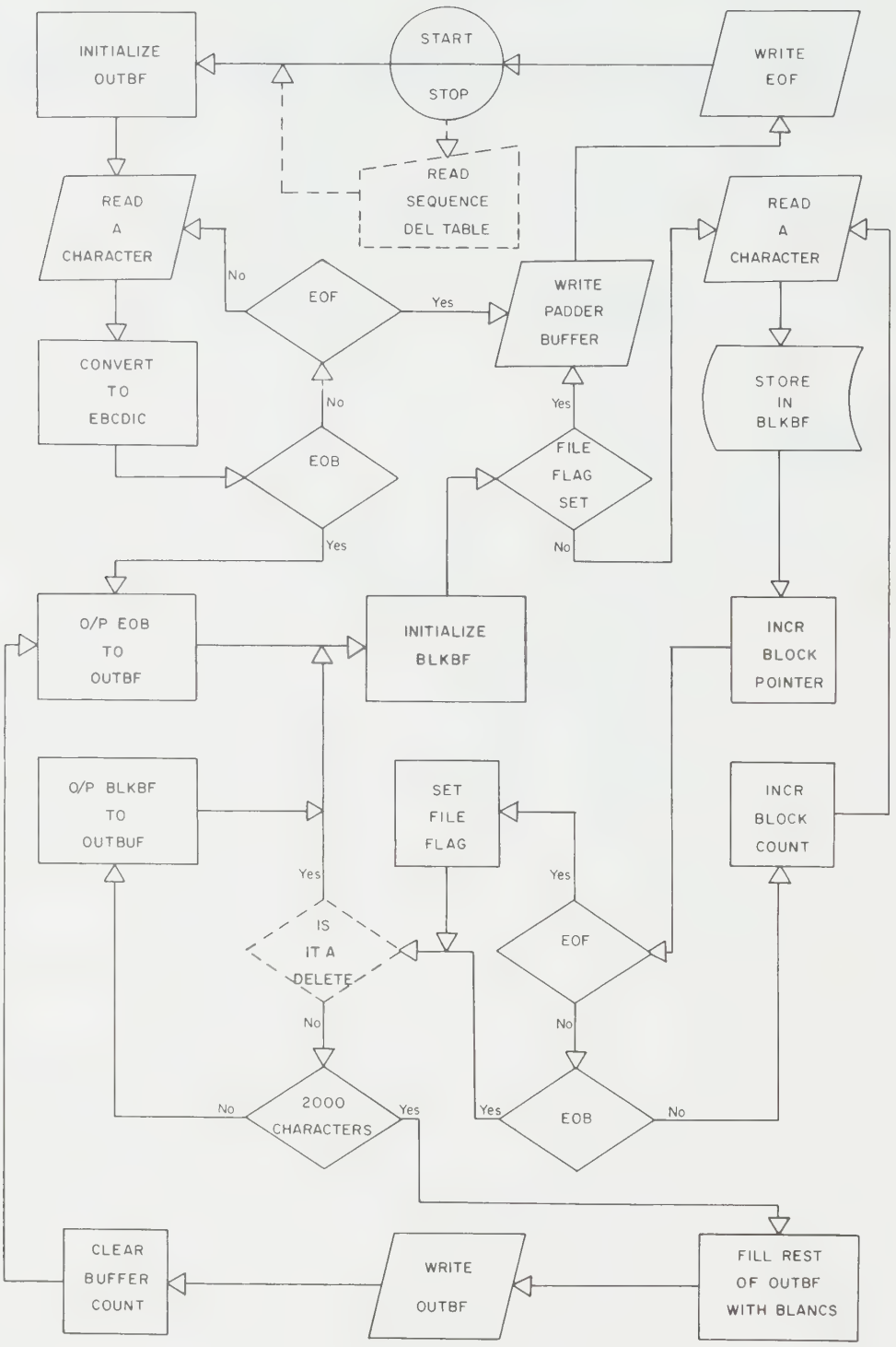
- each run searches one file mark

KEY IN OPTIONS:

1. to alter 9-track record length change
   35440 to oct xxxx(positive)

2. to enable sequence delete, load SEQDL patch thru ABL (requires program GERPL in core)
   - program first asks for sequence delete table in format
     code, subcode(i), subcode(n)
     note: code alone deletes every point maximum number of deletes = 33 end table with %, continues

3. FLOWCHART FOR 7 TO 9 TRACK CONVERSION



APPENDICES TO SECTION III

1. ECL For Gridding Package

2. Gridding Package Flowchart

1. ECL FOR GRIDDING PACKAGE

1. DIGITIZED DATA PROGRAM AND GRIDDING PROGRAM

```
@RUN
@ASG,T        GDATA.,F2/2/PØS/10
@ASG,T        17.,F2/2/PØS/10
@ASG,T        10.,F2/2/PØS/10
@ASG,T        11.,F2/2/PØS/10
@ASG,TJH      INFILE.,16N////EBCDIC// 6,XXXX
@ASG,TJV      GTP.,16N,YYYY
@XQT          .GDAT
@XQT          .GPLØT
@CØPY,GM      17.,GTP
@FIN
```

INFILE contains the digitized data. GDAT is the digitized data program which produces the coordinates file GDATA. This file is used by the gridding program GPLØT, to build a direct access file on unit 10, this direct access file is transferred to a sequential file on unit 17 before being copied to tape GTP. If the map being processed has boundaries within boundaries, then unit 11 is used to contain the switch matrix and GPLØT1 is executed instead of GPLØT.

2. MERGE PROGRAM

```
@RUN
@ASG,T        10.,F2/2/PØS/10
@ASG,T        11.,F2/2/PØS/10

@ASG,T        NN.,F2/2/PØS/10

@ASG,T        9.,F2/80/PØS/100
@ASG,TJV      IN.,16N,XXXX
@CØPY,G       IN.,10.
@CØPY,G       IN.,11.

@CØPY,G       IN.,NN.
@FREE         IN.
@XQT          .MRGE
@ASG,TJV      ØTP.,16N,YYYY
@CØPY,GM      9.,ØTP
FIN
```

As many areas as are required to contain all the separate maps are assigned as units 10 to NN. File IN contains the data from the gridding program. The files are merged onto unit 9 and then copied to ØTP.

3. SORT PROGRAM

```
@RUN,
@ASG,TJV      INP.,16N,XXXX
@ASG,T        22.,F2/80/PØS/100
@ASG,T        23.,F2/80/PØS/100
@CØPY,G       INP.,22
@FREE         INP.
@XQT          .FSØRT
 FSRTV,962,5,5,1,8,9,U,A,2,11,12,U,A
@ASG,TJV      ØTP.,16N,YYYY
@CØPY,GM      23.,ØTP
@FIN
```

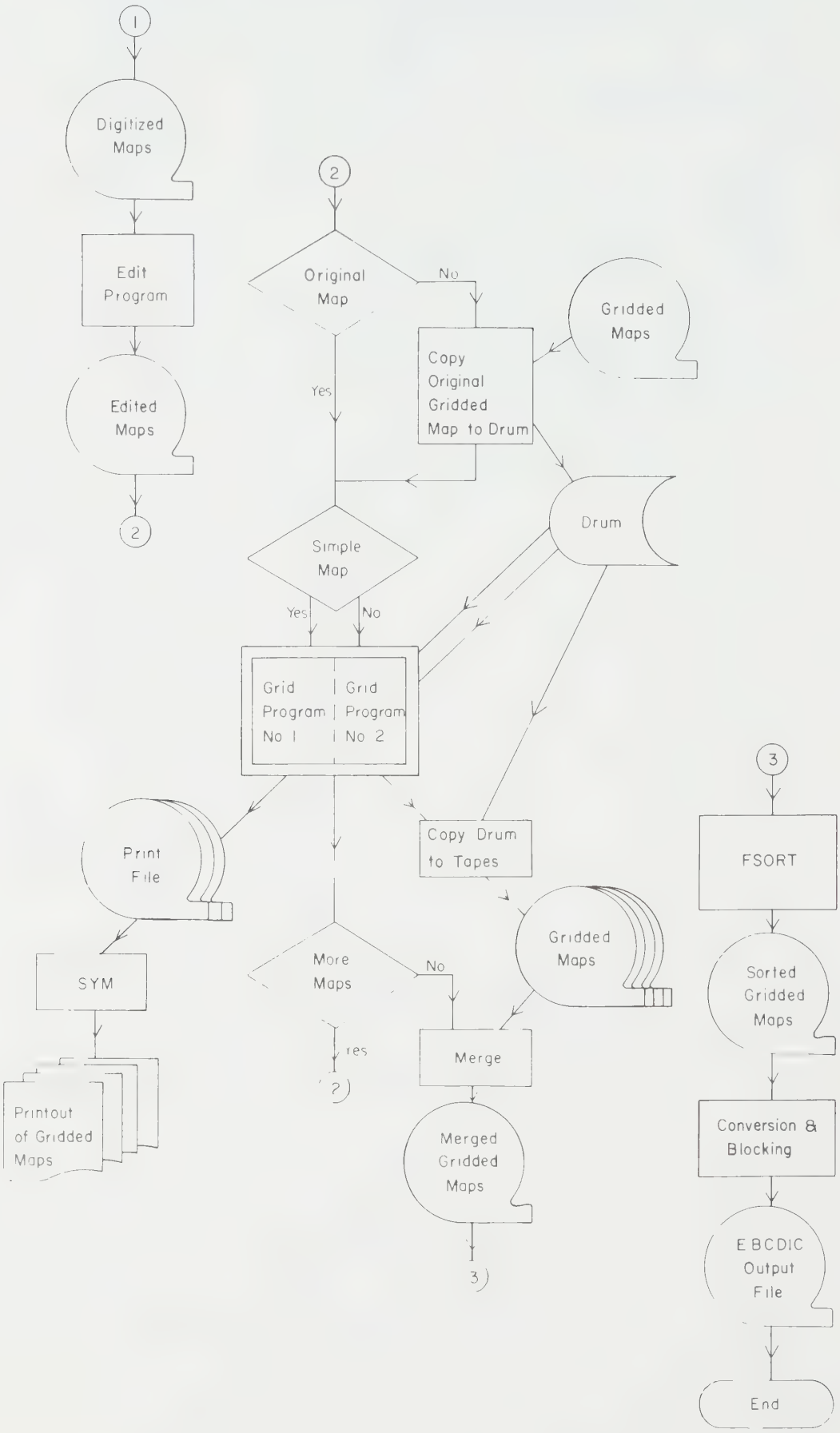
INP is the output file from the merge program. The sort is executed as a disk sort and the final output on unit 23 is then copied to tape ØTP.

4. CONVERSION PROGRAM

```
@RUN,
@ASG,TJV    IN.,16N,XXXX
@ASG,T      23.,F2/80/PØS/100
@CØPY,G     IN.,23.
@ASG,TJV    ØUTAPE.,16N////EBCDIC,YYYY
@XQT        .FTAPE
@FIN
```

IN contains the output from the sort program, this is copied into mass storage. The program FTAPE uses this file to produce the tape file ØUTAPE.

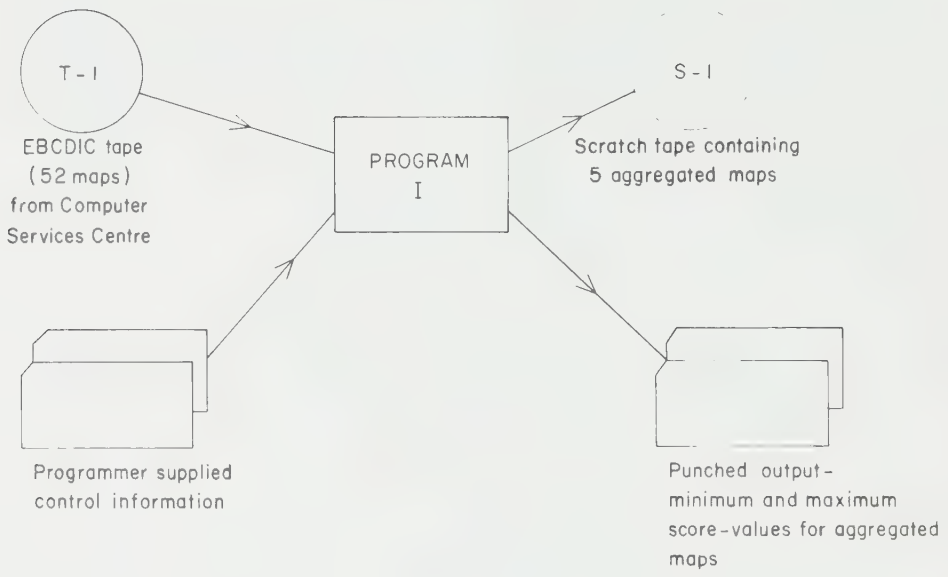
2. GRIDDING PACKAGE FLOW CHART

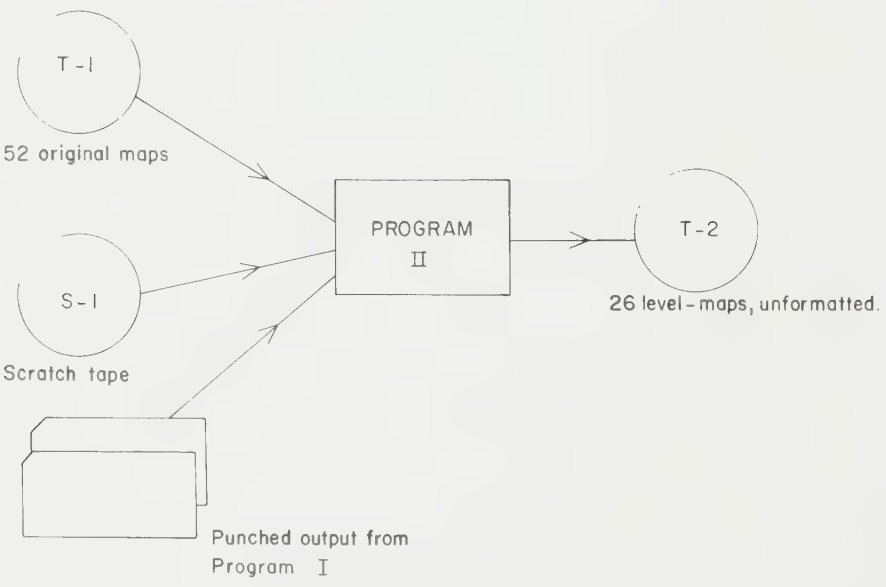


APPENDICES TO SECTION IV

1. Task I: Clean-up and Data Aggregation Flowchart
2. Task II: Data Dump
3. Task II: Data Dump Flowchart
4. Task III: User-Instruction Preprocessor
5. Task III: User-Instruction Preprocessor Flowchart
6. Task IV: Execute and/or Delete Program
7. List of Valid Map Numbers for the "Add" Command
8. List of Township Numbers, Names and Boundaries

TASK I: CLEANUP AND DATA AGGREGATION FLOW CHART





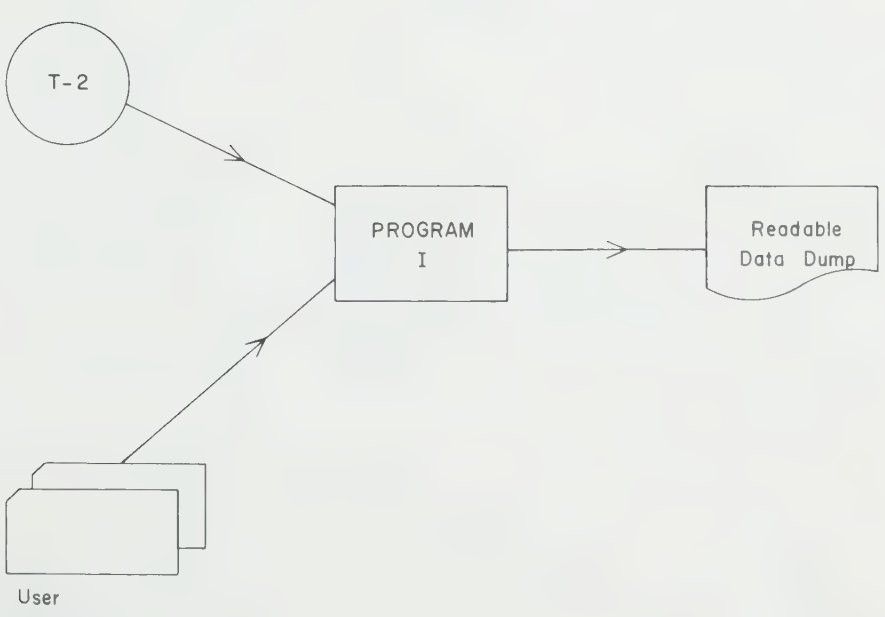
Appendix 2
Section IV

TASK II DATA DUMP

Commands	SubCommands	Action Taken
TITLE	followed by one card of text	-identifies output
PRINT	followed by N1, N2, N3, N4, END where the Ni are xmin, xmax, ymin, ymax respectively	-limits dump to specific rectangular window
STOP	Signals end of user input	
* (text)	comment card	

Appendix 3
Section IV

TASK II: DATA DUMP FLOW CHART



Commands	SubCommands	Action Taken
(text)	None	-start new package -place all 80 columns of this card into RUNID Variable.
*(Text)	None	-comment card, listed only and has no effect on program.
COPY	Followed by one of these = 1. OLD 2. STANDARD or 3. NEW If NEW specify package # N on next card (say package #10) 10 END	-OLD -brings in a copy of last package assembled error-free -STANDARD -brings the standard package into core -NEW -copies package # coded on last card.
WINDOW	followed by one of the following = 1. OLD 2. STANDARD or 3. NEW if NEW specify "RECTANGLE" or "TOWNSHIP" on next card if RECTANGLE, specify xmin, xmax, ymin, ymax, END on the next card. if TOWNSHIP, specify N, END where N is the township #, on next card.	Old and Standard copy in the window parameters from OLD AND STANDARD packages respectively whereas NEW enables user to specify his own Parameters.
SYMBOLS	followed by one of the following = 1. OLD 2. STANDARD or 3. NEW if NEW specified, provide symbols as an unpunctuated character string followed by END, all on one card.	-the number of characters in the string is stored in SYMLGT, the character string itself in SYMBLS (max 76)
GRADATIONS	followed by one of = 1. OLD 2. STANDARD or 3. NEW if NEW specified, provide on next card N1, N2, END where N1 = # of overprints desired, N2 = length of grey-scale symbolism, followed by N1 cards showing the grey-scale.	-N1 is stored in OVRPRT N2 is stored in GRDLGT the N1 cards are stored in array GRDTMP (20, 4) N1 must not exceed 4, N2 must not exceed 80.

TASK III USER-INSTRUCTION PREPROCESSOR

Commands	SubCommands	Action Taken
RANGE	Followed by one of these: 1. OLD 2. STANDARD 3. NEW If NEW is used, it is followed by card with N1, N2, N3,...Nm, END where Ni is the relative spread of m classes specified.	-the Ni are scaled so that their sum equals 100, and are placed in array RANGT, m is stored as RNGLGT. RNGLGT must be less than or equal to GRDLGT and to SYMLGT.
WEIGHTS	Followed by one of these: 1. OLD 2. STANDARD 3. NEW If NEW is used, it is followed by cards of the form Column 1 M..... W..... END (fld 1) (fld 2) . . . terminated by MEND or END or M99999 where fld 1 is for map # (integer), fld 2 is for weight (integer or decimal number).	
RATINGS	Followed by one of these: 1. OLD 2. STANDARD 3. NEW If NEW is used, it is followed by cards of the form: 1 7 13 19 M..... L..... R..... END (fld 1) (fld 2) (fld 3) when fld 1 is for map #(integer) 2 is for level(integer) 3 is for rating(integer or decimal number) terminated as in WEIGHTS	

TASK III USER-INSTRUCTION PREPROCESSOR

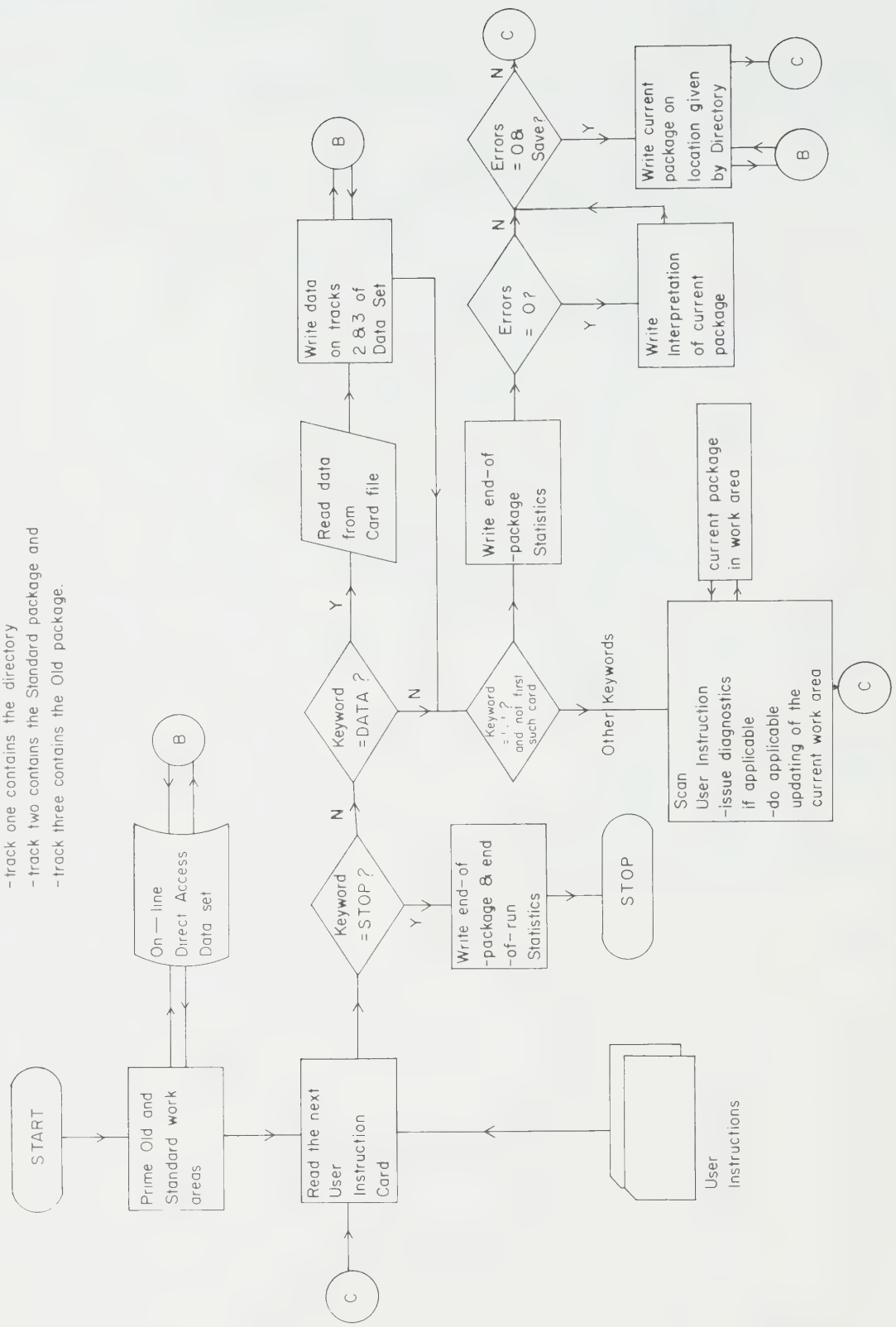
Commands	SubCommands	Action Taken
CCOPY	followed by one of the following = 1. OLD 2. STANDARD and followed by composite #'s to be copied -CCOPY can be used twice, once for OLD and once for STANDARD. followed by N1, N2, ... Nm, END Where the Ni are the composite numbers to be copied from the OLD or the STANDARD packages. No Ni should exceed 10.	-ccopy stands for composite copy.
COMPOSITE	followed by any of the following commands	-start new composite definition.
TITLE	followed by text desired on the next card (map identification for this composite)	-puts text card into MAPID of current composite.
ADD	followed by map #'s N1, N2,...Nm, END, where N1, N2...Nm are valid map #'s as listed in Appendix 6.	-looks up relative positions of these maps on tape and overlays them to generate a score map.
MASK	followed by max. 10 cards as follows = Column 1 7 11 34 11 56 M..... L..... EQ .R (fld 1) (fld 2) NE and terminated by MEND or END or M00000 where fld 1 is meant for map #, fld 2 for level on that map. Col 13, 14 should have the logical operator = EQ-for the level on current composite to be equal to the level in fld 2, or NE-for not equal. Col 15 contains the replacement character to be used if the relationship is satisfied.	

TASK III USER-INSTRUCTION PREPROCESSOR

Commands	SubCommands	Action Taken
USAGE	followed by one of the following: 1. DISPLAY - display the composite only 2. DATUM - display the composite and store it to serve as basis for a later compare or correlate 3. COMPARE - with the last datum saved 4. CORRELATE - specified levels of this composite with the levels specified for datum. 5. DCOMPARE - same as 3 and also store this composite as the new datum. The old datum is over-ridden. 6. DCORRELATE - same as 4 and also store this composite as the new datum. The old datum is over-ridden.	
PRINT	followed by either 1. SYMBOLS or 2. GRADATIONS	-the composite according to symbols/gradation as specified.
STATISTICS	followed by: N1, N2,..., Nm, END Where the Ni are map #s against which the statistics collection is desired.	
STOP	Signals end of user input - is the last card in user input.	

TASK III : USER INSTRUCTION-PREPROCESSOR FLOW CHART

Notes on Direct Access Data Set
- track one contains the directory
- track two contains the Standard package and
- track three contains the Old package.



Appendix 6
Section IV

TASK IV EXECUTE AND/OR DELETE PROGRAM

Commands	SubCommands	Action Taken
EXECUTE	N, END where N is the package # to be executed. Only one # may be specified.	
DELETE	followed by N1, N2,.Nm, END	deletes references to the m (specified) packages from the directory, effectively freeing space for new packages.

Appendix 7
Section IV

LIST OF VALID MAP NUMBERS FOR THE " ADD " COMMAND

102	Aircraft Noise
103	Air Pollution
104	Agricultural Capability
203	Accessibility to Sudbury
205	Local Services
206	Water Supply Systems
207	Sewage Disposal Systems
208	Highway Cost Capacity
209	Railway Accessibility
210	Accessibility to Copper Cliff
211	Accessibility to Levack
212	Accessibility to Falconbridge
214	Ground Water Environments
401	Wildlife Composite
402	Fisheries Composite
403	Scenic Composite
404	Recreation Composite
405	Vegetation Composite

LIST OF TOWNSHIP NUMBERS, NAMES AND BOUNDARIES

OWNSHIP NUMBER	NAME	XMIN	XMAX	YMIN	YMAX
1000	SUDBURY REGIONAL MUN.	45	308	50	288
2000	ORGANIZED MUNICIPALITY	18	420	1	175
3000	UNORG. MUNICIPALITY	2	465	1	288
4000	INDIAN RESERVES	2	275	20	260
1001	WALDEN	45	200	50	170
1002	SUDBURY	194	253	105	180
1004	ONAPING FALLS	87	131	150	232
1005	RAYSIDE-BALFOUR	125	200	150	232
1006	VALLEY EAST	160	270	170	260
1007	CAPREOL	194	270	218	288
1008	NICKEL CENTRE	230	308	140	240
2001	NAIRN	18	57	80	116
2002	HAGAR	340	383	140	175
2004	RATTER & DUNNET	378	420	110	175
2005	CASIMIR J. & A.	340	420	75	150
2006	COSBY, M. & M.	340	420	1	60
3001	ULSTER	10	57	255	288
3002	MUNSTER	52	95	255	288
3003	LEINSTER	87	131	255	288
3004	TYRONE	125	168	255	288
3005	KITCHENER	160	200	255	288
3006	PARKIN	230	270	255	288
3007	AYLMER	266	307	255	288
3008	MACKELCAN	303	344	255	288
3009	PART MCCARTHY	340	360	255	288
3010	MONCRIEFF	18	57	227	260
3011	HESS	52	95	227	260
3012	HARTY	87	131	227	260
3013	FOY	125	168	227	260
3015	RATHBUN	266	344	227	260
3016	PART KELLY	340	360	227	260
3017	SCADDING	303	344	197	232
3018	PART DAVIS	340	360	197	232
3019	PART LOUGHRIN	340	360	170	204
3020	STREET	303	344	170	204
3021	AWREY	303	344	140	175
3022	PART DRYDEN	266	307	140	150
3023	PART DILL	245	270	110	150
3024	CLELAND	266	307	110	150
3025	HAWLEY	303	344	110	150
3026	HENDRIE	303	344	80	116
3027	BURWASH	266	307	80	116
3028	CHERRIMAN	340	383	50	87
3029	HADDO	378	420	50	87
3030	HOSKIN	303	344	50	87
3031	DELAMERE	303	344	20	60
3032	SERVOS	266	307	50	87
3033	SECORD	230	270	80	116
3034	TILTON	194	235	80	116
3035	LAURA	230	270	50	87
3036	HALIFAX	194	235	50	87
3037	EDEN	160	200	80	116
3038	BEVIN	160	200	50	87
3039	CAEN	125	168	50	87
3041	COX	266	307	20	60
3042	WALDIE	230	270	20	60
3043	ATTLEE	194	235	20	60
3044	SALE	160	200	20	60
3045	GOSCHEN	125	168	20	60
3046	STALIN	87	131	20	60
3047	ROOSEVELT	52	95	20	60
3048	CUTIN	2	57	20	60
3049	TRUMAN	52	95	50	87
3050	FOSTER	18	57	50	87
3051	HYMAN	18	57	110	150
3052	TOTTEN	18	57	140	175
3053	PART TRILL	52	95	150	175
3054	CASCADEN	52	95	170	204
3055	ERMATINGER	18	57	170	204
3056	CARTIER	52	95	197	232
3057	HART	18	57	197	232
3060	ALLEN	266	307	1	30